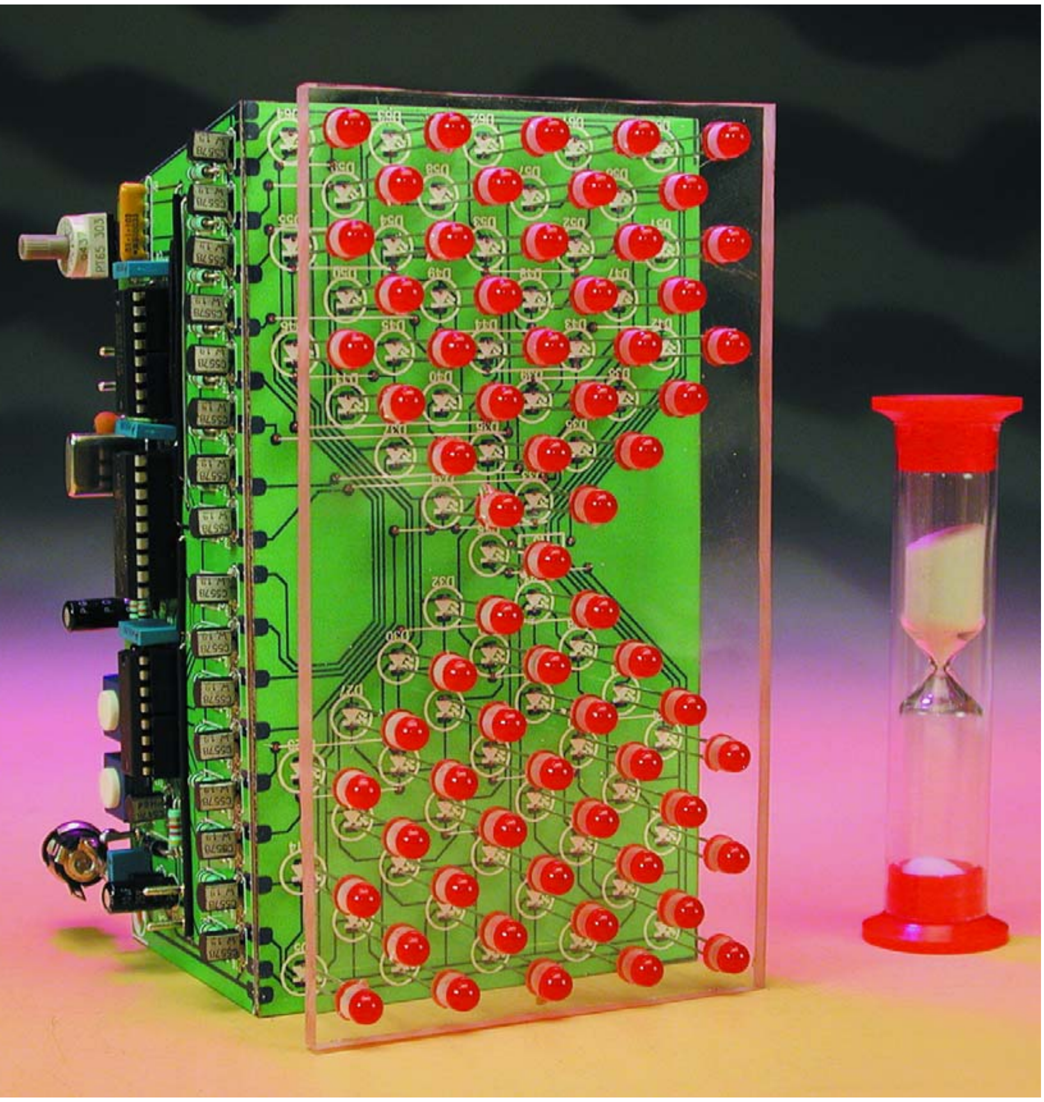
Electronic Hourglass

an unusual PIC project

Design by S. Buchgeher

In a real hourglass, silicon dioxide (SiO_2) in the form of extremely fine quartz sand trickles from one bulb into the other. In the electronic version, the only silicon to be found is in a small, oscillating wafer of quartz and a few chips in the PIC microcontroller and transistors. And instead of grains of sand, points of light trickle downwards…



Of course, the points of light in the electronic hourglass do not have any direct relationship to the silicon in a conventional hourglass, since there is no silicon among the various elements (primarily gallium and arsenic) in the chips of the LEDs used here. Aside from that, you might wonder why anyone would want to build an electronic hourglass.

One reason is that this project provides an excellent example of how to program and use a PIC microcontroller. By studying the listings (which can be downloaded from the *Elektor Electronics* website), you can see how the program is constructed, and if you wish you can adapt the parameters configured in the software to suit your own needs.

Another reason is that the electronic hourglass also serves a practical purpose, since it can be used as a kitchen timer that has at least three advantages over a traditional hourglass. First, the time required for the ‘sand’ to flow from one bulb to the other can be adjusted; second, there is an acoustic signal after the time is expired; and third, the whole thing looks great and fits better in the household of a modern electronic hobbyist. Of course, the fact that it’s somewhat unique (since you can’t just buy it ready-made) is also a significant consideration.

The hardware

The most striking part of the schematic diagram (**Figure 1**) is the 8 × 8 matrix of LEDs, which represents one of the key elements of this design. But what happened to the series resistors for the LEDs? Here the LEDs are driven using the multiplexing technique, so each LED is on for only a fraction of a second. This means that we can do without series resistors. Each of the rows and columns of the 8 × 8 matrix is driven via a transistor stage. Two 3-to-8 decoders (IC2 and IC3) ensure that only one row or column at a time is enabled. The LEDs that should be illuminated at any given time are determined by the microcontroller (IC1). This brings us to the second major part of the circuit, the PIC16F84 microcontroller. This device is readily available and has exactly the number of I/O pins needed for our application. The main advantage of this

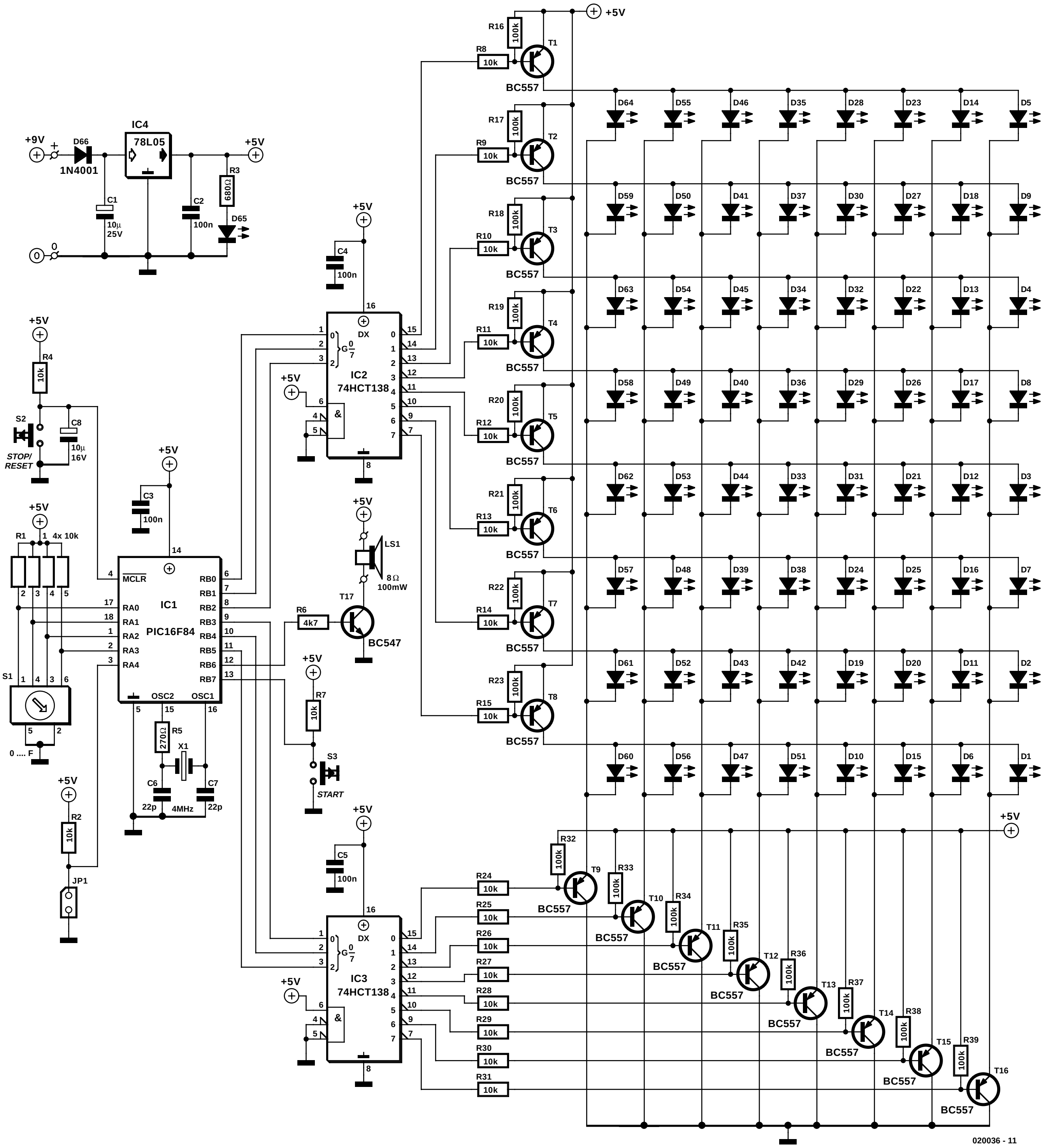


Figure 1. The circuit of the electronic hourglass essentially consists of PIC microcontroller driving a display formed by 64 LEDs

microcontroller is that it has a Flash memory, so it can be reprogrammed practically as often as desired. There is also an EEPROM in the PIC16F84, but it is not needed for this project.

The job of the microcontroller is to control the entire process, from starting the timer to emitting an acoustic signal at the end of the set interval. This is realised using the program described later on.

A standard circuit connected to pins 15 and 16 of the microcontroller is used to generate the clock signal. In principle, a 4-MHz ceramic resonator could be used in place of the quartz crystal (X1), but the times would then be less exact. If a quartz crystal is used, the configured values (timing constants) are sufficiently close to the to the calculated values.

A proven standard solution using a resistor and capacitor (R4 and C8) is used to generate the reset signal. The Stop switch (S2), which is wired in parallel with C8, also generates a reset signal to put the microcontroller back into its initial state.

The row decoder (IC2) is connected to I/O pins RB0–RB2, while the column decoder is connected to pins RB3–RB5. These six I/O pins are thus responsible for driving the LED matrix. The LEDs are 5-mm, red low-current types (2 mA typical). Of course, the circuit will also work with green or yellow LEDs, provided low-current types are used with sufficient brightness.

The loudspeaker (LS1) is driven by pin RB6 of the microcontroller via a transistor stage (R6/T7), using a highly audible square-wave signal.

The desired time is set using a hexadecimal-coded switch (S1), allowing 16 different times to be set. This switch requires four lines, each with its own 10-kΩ pull-up resistor array (R1). Microcontroller pins RQ0–RA3 are used for this purpose.

A tone can optionally be generated each time the state of the LEDs changes by fitting jumper JP1 (connected to pin RA4 of the microcontroller).

The LED hourglass is started using the Start pushbutton switch S3, which is connected to pin RB7 of the microcontroller via a pull-up resistor (R6).

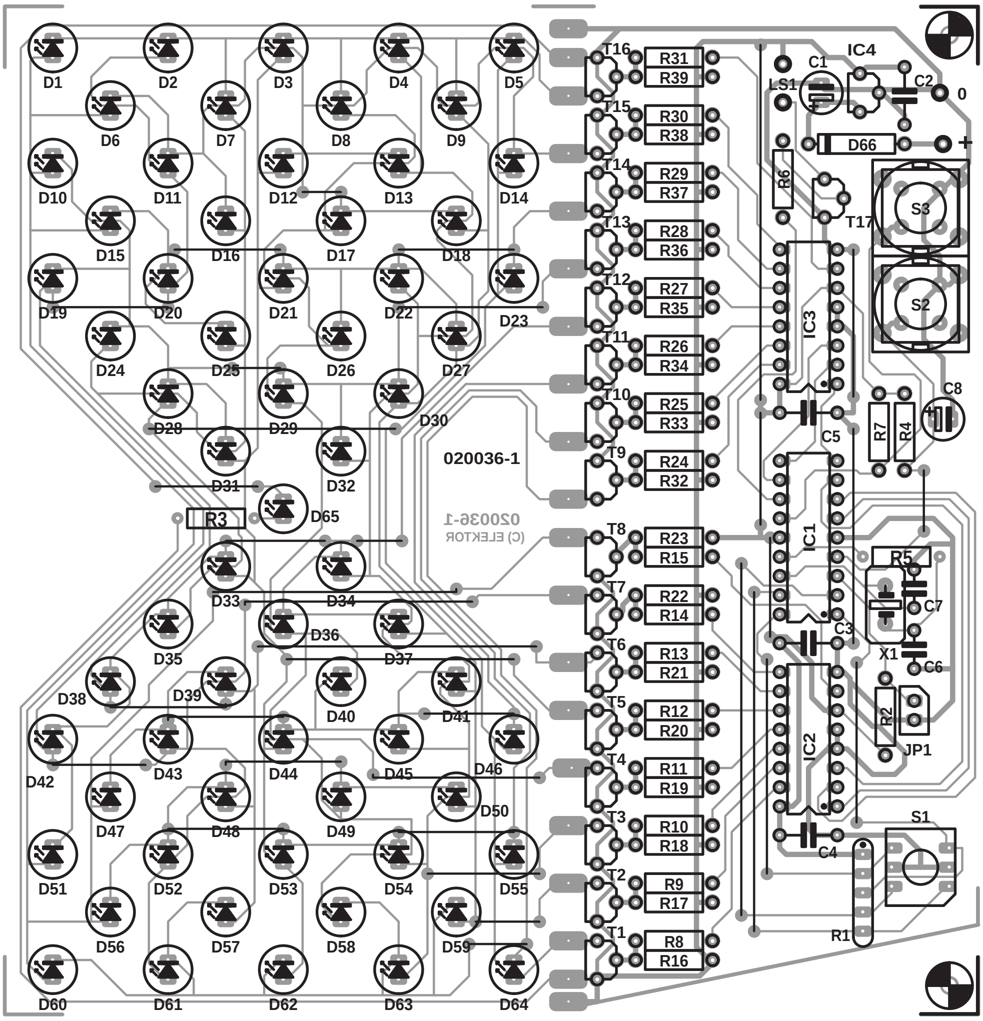


Figure 2. The layout of the printed circuit board betrays its intended use: the LEDs are

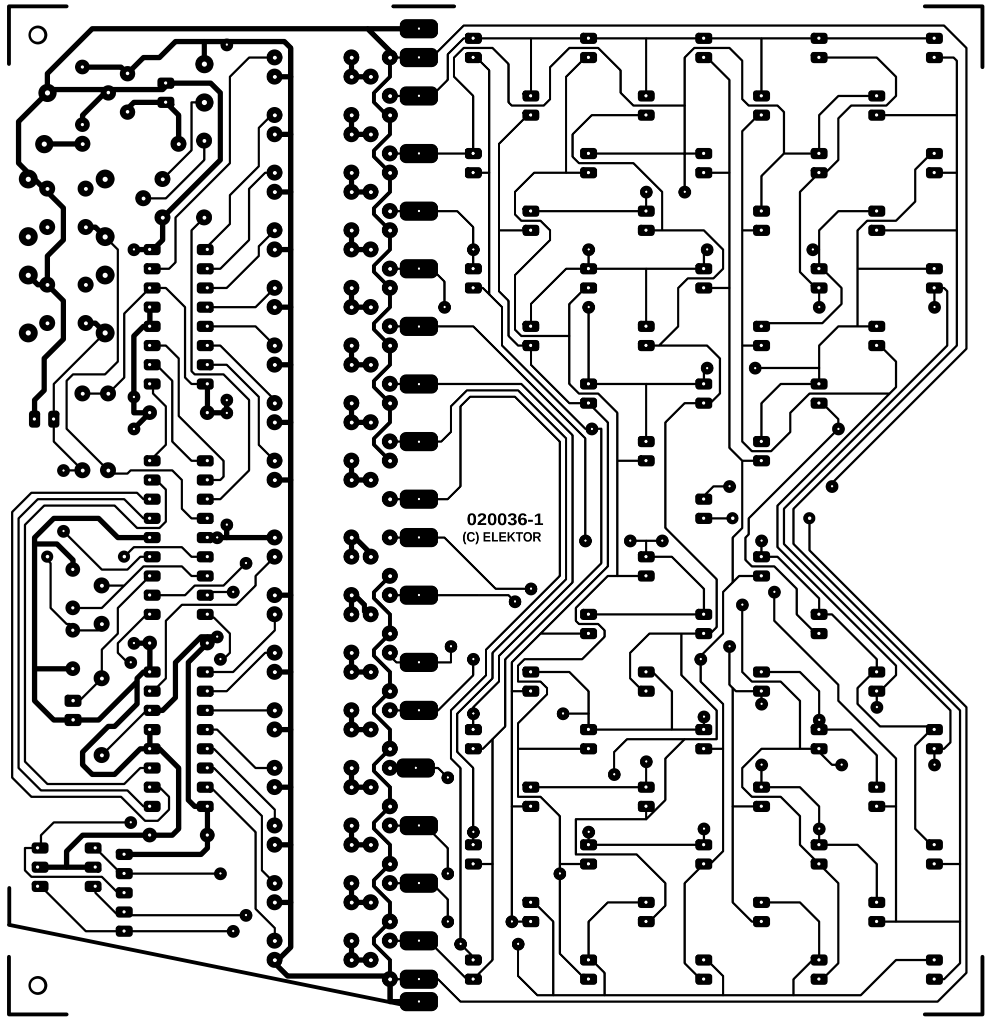
COMPONENTS LIST

Resistors:
R1 = 4-way 10kΩ SIL array
R2, R4, R7-R15, R24-R31 = 10kΩ
R3 = 680Ω
R5 = 270Ω
R6 = 4kΩ7
R16-R23, R32-R39 = 100kΩ

Capacitors:
C1,C8 = 10µF 25V radial
C2-C5 = 100nF
C6,C7 = 22pF

Semiconductors:
D1-D65 = LED, 5mm, red, low-current
D66 = 1N4001
T1-T16 = BC557
T17 = BC547
IC1 = PIC16F84-04/P, programmed, order code **020036-41**
IC2, IC3 = 74HCT138
IC4 = 78L05

Miscellaneous:
X1 = 4MHz quartz crystal
S1 = miniature hex encoder switch, 16 positions (Hartmann type PT65 303, from Conrad Electronics, # 705497)
S2,S3 = single contact switch, make contact, e.g., type D6
Ls1 = miniature loudspeaker, 8Ω 0.1W
JP1 = jumper
PCB, order code **020036-1**
Disk, project software, order code **020036-11** or Free Download



arranged in the form of an hourglass (board available ready-made).

LED D65 is connected directly to the supply voltage via a series resistor (R3), so it is always on when the supply voltage is present. It is placed in the middle of the 'LED field' to act as a pilot light indicating that the hourglass is switched on.

There's not much that needs to be said about the power supply. A 78L05 fixed voltage regulator, in combination with capacitors C1 and C2, provides a stabilised 5 V operating voltage. Diode D1 protects against reverse polarity, and an on/off switch on the 9-V input (not shown on the drawing) allows the hourglass to be switched on and off. The 9-V source can be a 9-V mains adapter, a 9-V battery pack (six 1.5-V cells) or a 9-V storage battery, as desired. The current consumption of the circuit is approximately 45 mA.

Construction and operation

All of the components shown in the schematic diagram can be seen in the component layout of the printed circuit board (**Figure 2**). The 65 LEDs are arranged to form a display in the shape of a traditional hourglass. The board can be split into an LED board and a control board (containing the rest of the circuitry) if you do not want to have the control portion immediately adjacent to the LED display. The separation line runs through the middle of the row of solder pads between the two parts of the circuit board. The two halves of the solder pads remaining on each board after they have been separated are used to attach the interconnecting wiring. For the prototype built in the *Elektor Electronics* lab, the two parts of the board were arranged at right angles to each other (see photos). This allows the connections to be directly soldered together (using short lengths of wire if desired).

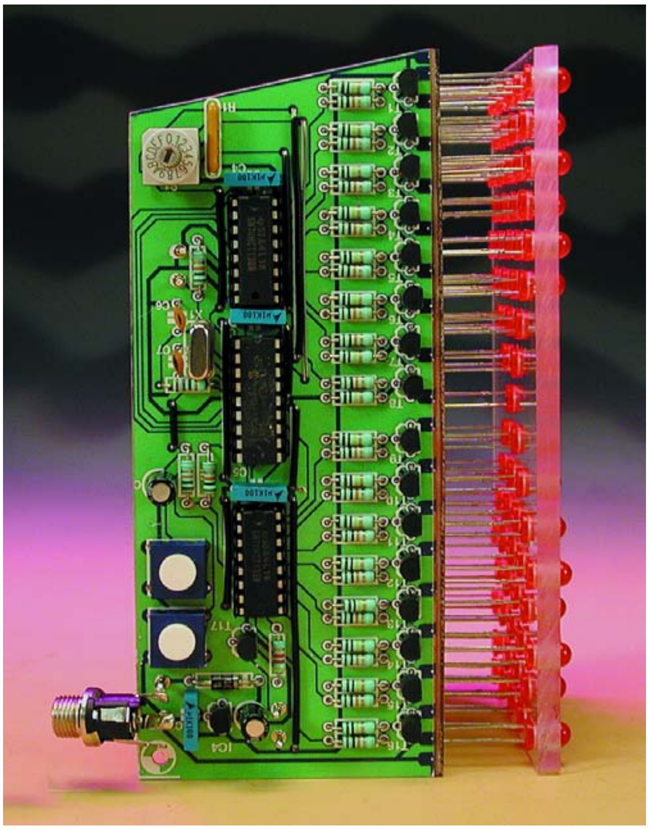
A few suggestions, both general and specific, are in order with regard to fitting the components to the circuit board. Start by fitting the wire bridges, to ensure that you do not forget any of them. Pay attention to the polarity of polarised components, such as electrolytic capacitors, diodes and LEDs. Use sockets for IC1–IC3, and be sure to insert the ICs the right way around (note the markings). Before connecting the operating voltage, thoroughly inspect the fitted components and solder joints of the assembled board. In the Components List, no current gain class is given for the transistors, so A, B or C types can be used.

There is no on/off switch on the circuit board. Such a switch can be fitted to the rear of the enclosure, with the controls (the Start and Stop pushbuttons and the 16-position code switch) at the front, fitted directly to the circuit board or on a front panel.

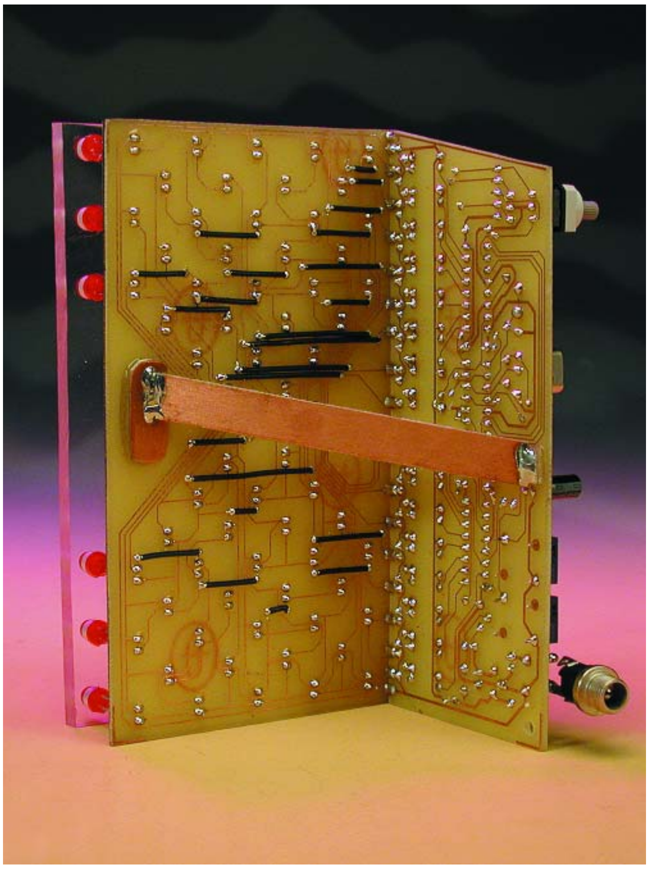
After you apply the operating voltage and switch on the circuit using the on/off switch (if present), the lower group of LEDs will light up. Set the desired time interval using the selector switch. You can choose from 16 intervals, ranging from 15 seconds to 1 hour (see **Table 1**). The intervals are defined in the software using two constants, so they can be modified if necessary.

As soon as the Start button is pressed, the LEDs in the upper half of the display will light up and the set time will start to trickle away. The LEDs in the upper part of the display will be extinguished one by one, while the LEDs in the lower part of the display will be correspondingly illuminated one by one. The timer interval can be terminated at any time by pressing the Stop button.

All of the LEDs in the lower half of the display will be illuminated when the set interval has expired. When the last LED lights up, the loudspeaker (fitted at the rear) starts to emit a tone. This signal can be stopped by pressing the Stop button. If you press the Start button instead of the Stop button, the loudspeaker



will also be muted, but at the same time the hourglass will be started again to time out the interval set by the code switch.



If the previously mentioned jumper is fitted to the board, a brief acoustic signal will be emitted each time the state of the LEDs changes.

The software

The tasks performed by the microcontroller can essentially be divided into the following subtasks:

– Driving the 8 × 8 LED matrix.
This should take place in the background, so it is implemented using a timer interrupt.

– Loading the addresses of the LEDs to be illuminated into memory locations POS1–POS32.

– Polling the Start pushbutton and then the hex-encoded switch. Separate polling of the Stop switch is not necessary, since it is directly connected to the Reset input. A reset is triggered when the Stop button is pressed, causing the microcontroller to be re-initialised.

– Activating the loudspeaker on completion of the timing interval.

The special feature of the software for this project is that the addresses of the 32 LEDs that are illuminated at any given time are stored in a set of RAM locations labelled POS1 through POS32. In the initial state, with the 32 bottom LEDs illuminated, RAM locations POS1 through POS32 contain the addresses of LEDs D33 through D64. POS1 thus contains the binary value ‘10011100’ (the address of LED D33), POS2 contains the binary value ‘10010100’ (the address of LED D34) and so on.

Table 1.

Time settings selectable using the code switch S1.

Switch setting	Time	
0	15	seconds
1	30	seconds
0	60	seconds
3	90	seconds
4	2	minutes
5	3	minutes
6	4	minutes
7	5	minutes
8	8	minutes
9	10	minutes
A	12	minutes
B	15	minutes
C	20	minutes
D	30	minutes
E	45	minutes
F	60	minutes

The ISR is given the additional task of decrementing counter register AKTXBITx by 1 each time it is called. This register is a 16-bit counter register, so it must be divided into two 8-bit registers labelled AKTZEITL and AKTZEITH. When the value of this register reaches zero, a flag bit is set (NaechstZust-Flag) to inform the main routine that the state of the LEDs must be changed. Two subroutines, labelled ZUSTAND1 and ZUSTAND0, are provided for the start and exit states (the start state is with all 32 of the upper LEDs illuminated, while the exit state is with all 32 of the lower LEDs illuminated).

Although it would have been possible to write a subroutine for each of the intermediate states, the program memory of the PIC16F84 is probably not large enough to hold the amount of code this would generate. However, other solutions are possible — and if you are interested in the details, you can analyse the program by studying the listings.

(020036-1)